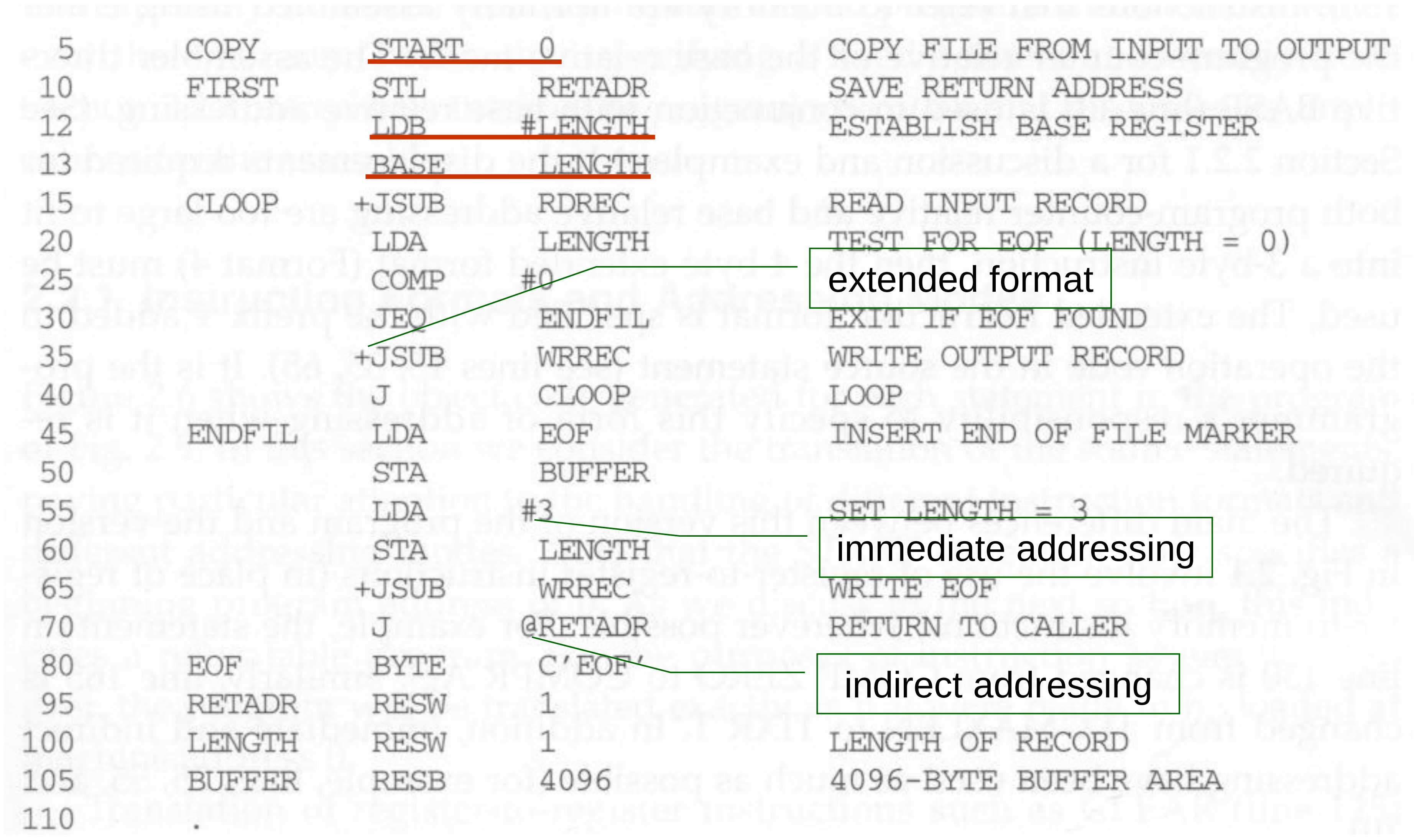
2.2 Machine-Dependent Assembler Features (SIC/XE Assembler)

Instruction Formats, Addressing Modes, and Program Relocation

SIC/XE Assembly Program(fig.2.5)



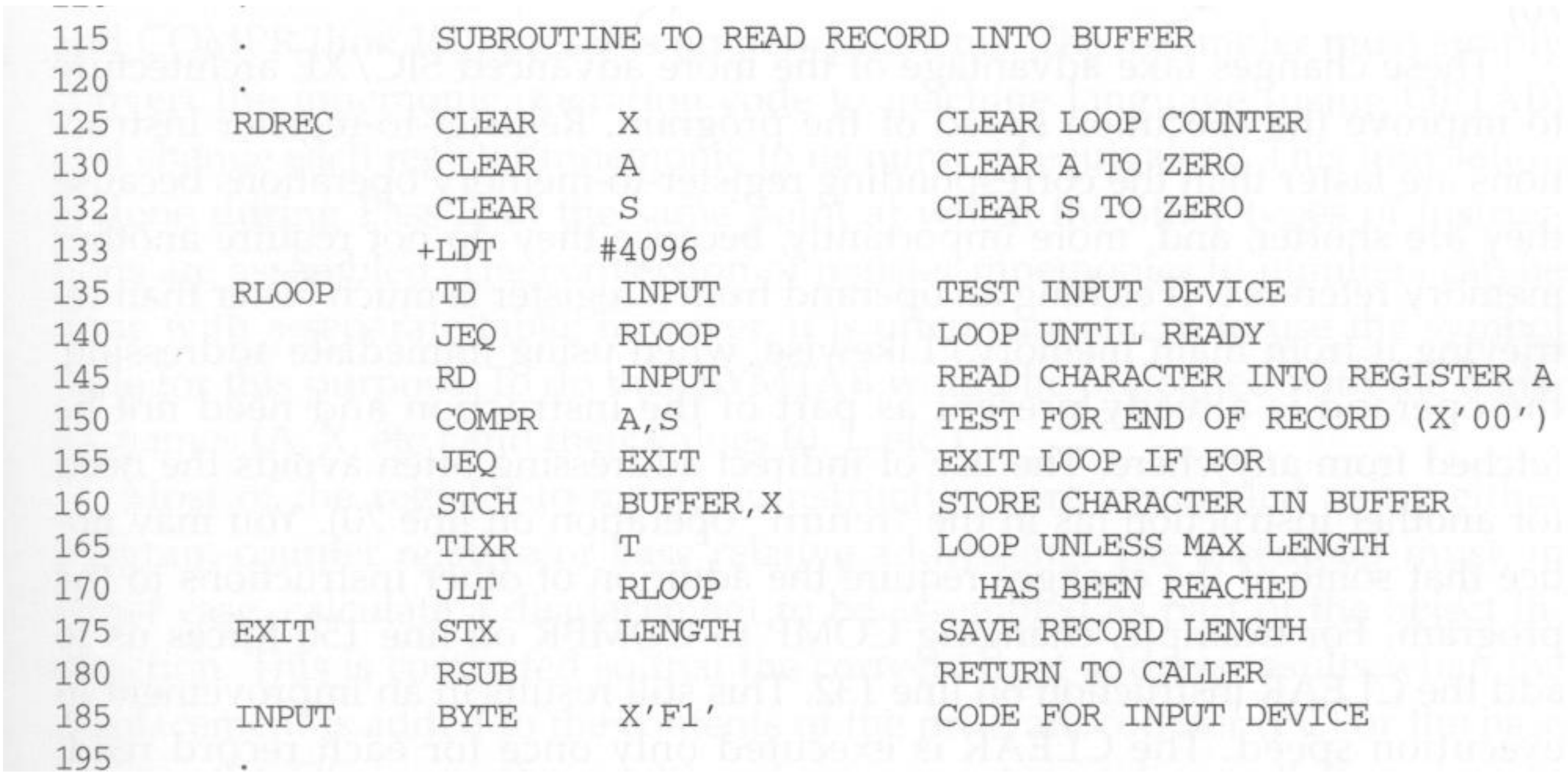
```
  5    COPY      START    0            COPY FILE FROM INPUT TO OUTPUT
 10    FIRST     STL      RETADR       SAVE RETURN ADDRESS
 12              LDB     #LENGTH       ESTABLISH BASE REGISTER
 13              BASE     LENGTH
 15    CLOOP    +JSUB     RDREC        READ INPUT RECORD
 20              LDA      LENGTH       TEST FOR EOF (LENGTH = 0)
 25              COMP    #0
 30              JEQ      ENDFIL       EXIT IF EOF FOUND
 35             +JSUB     WRREC        WRITE OUTPUT RECORD
 40              J        CLOOP        LOOP
 45    ENDFIL    LDA      EOF          INSERT END OF FILE MARKER
 50              STA      BUFFER
 55              LDA     #3            SET LENGTH = 3
 60              STA      LENGTH
 65             +JSUB     WRREC        WRITE EOF
 70              J       @RETADR       RETURN TO CALLER
 80    EOF       BYTE     C'EOF'
 95    RETADR    RESW     1
100    LENGTH    RESW     1            LENGTH OF RECORD
105    BUFFER    RESB     4096         4096-BYTE BUFFER AREA
110    .
```

extended format

immediate addressing

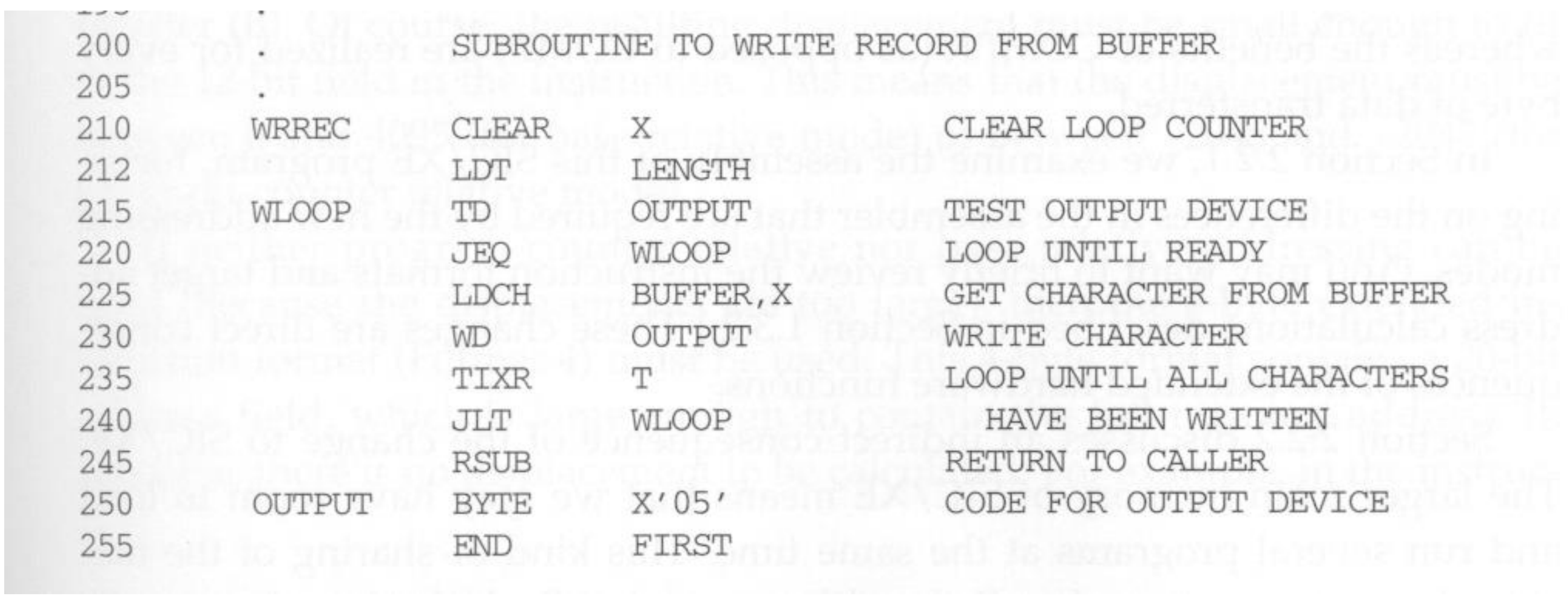
indirect addressing

SIC/XE Assembly Program



```
        .       SUBROUTINE TO READ RECORD INTO BUFFER
        .
RDREC   CLEAR     X               CLEAR LOOP COUNTER
        CLEAR     A               CLEAR A TO ZERO
        CLEAR     S               CLEAR S TO ZERO
       +LDT      #4096
RLOOP   TD        INPUT           TEST INPUT DEVICE
        JEQ       RLOOP           LOOP UNTIL READY
        RD        INPUT           READ CHARACTER INTO REGISTER A
        COMPR     A,S             TEST FOR END OF RECORD (X'00')
        JEQ       EXIT            EXIT LOOP IF EOR
        STCH      BUFFER,X        STORE CHARACTER IN BUFFER
        TIXR      T               LOOP UNLESS MAX LENGTH
        JLT       RLOOP             HAS BEEN REACHED
EXIT    STX       LENGTH          SAVE RECORD LENGTH
        RSUB                      RETURN TO CALLER
INPUT   BYTE      X'F1'           CODE FOR INPUT DEVICE
        .
```

SIC/XE Assembly Program



```
         .
         .       SUBROUTINE TO WRITE RECORD FROM BUFFER
         .
WRREC    CLEAR     X                 CLEAR LOOP COUNTER
         LDT       LENGTH
WLOOP    TD        OUTPUT            TEST OUTPUT DEVICE
         JEQ       WLOOP             LOOP UNTIL READY
         LDCH      BUFFER,X          GET CHARACTER FROM BUFFER
         WD        OUTPUT            WRITE CHARACTER
         TIXR      T                 LOOP UNTIL ALL CHARACTERS
         JLT       WLOOP               HAVE BEEN WRITTEN
         RSUB                        RETURN TO CALLER
OUTPUT   BYTE      X'05'             CODE FOR OUTPUT DEVICE
         END       FIRST
```

Benefits of SIC/XE Addressing Modes

- Register-to-register instructions
  - Shorter than register-to-memory instructions
  - No memory reference
- Immediate addressing mode
  - No memory reference. The operand is already present as part of the instruction
- Indirect addressing mode
  - Avoids the needs for another instruction
- Relative addressing mode
  - Shorten than the extended instruction
  - Easy program relocation

Considering Instruction Formats

- START directive specifies a beginning program address of 0: a relocatable program.
- Register-to-register instructions: simply convert the mnemonic name to their number equivalents
  - OPTAB: for opcodes
  - SYMTAB: preloaded with register names and their values

Considering Instruction Formats

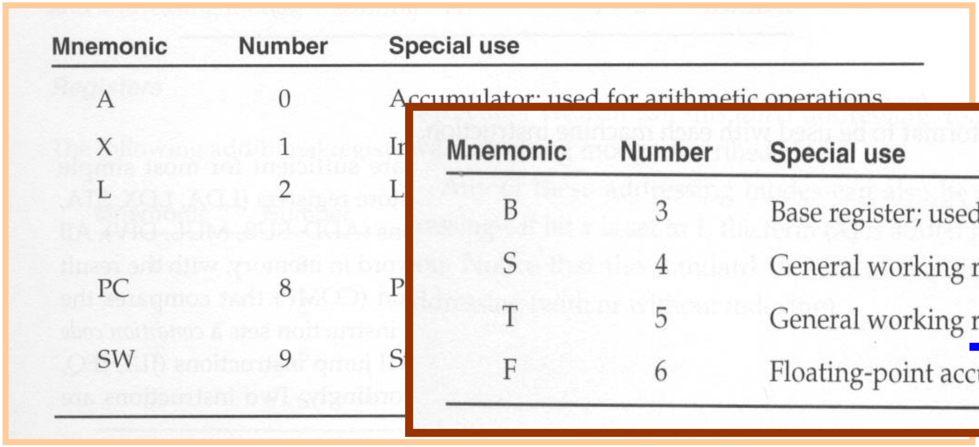
- COMPR A,S

```
- - - -  - - - -  - - - -  - - - -
```

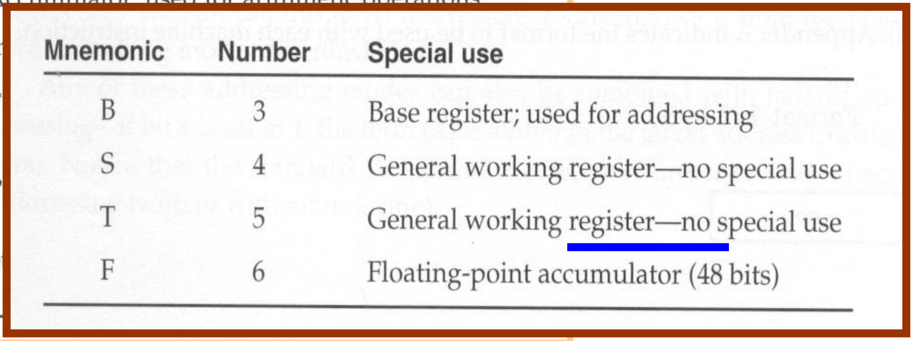
1010 0000 0000 0100 ➔ A004

- CLEAR X

1011 0100 0001 0000 ➔ B410



Mnemonic	Number	Special use
A	0	Accumulator; used for arithmetic operations
X	1	In
L	2	L
PC	8	P
SW	9	S



Mnemonic	Number	Special use
B	3	Base register; used for addressing
S	4	General working register—no special use
T	5	General working register—no special use
F	6	Floating-point accumulator (48 bits)

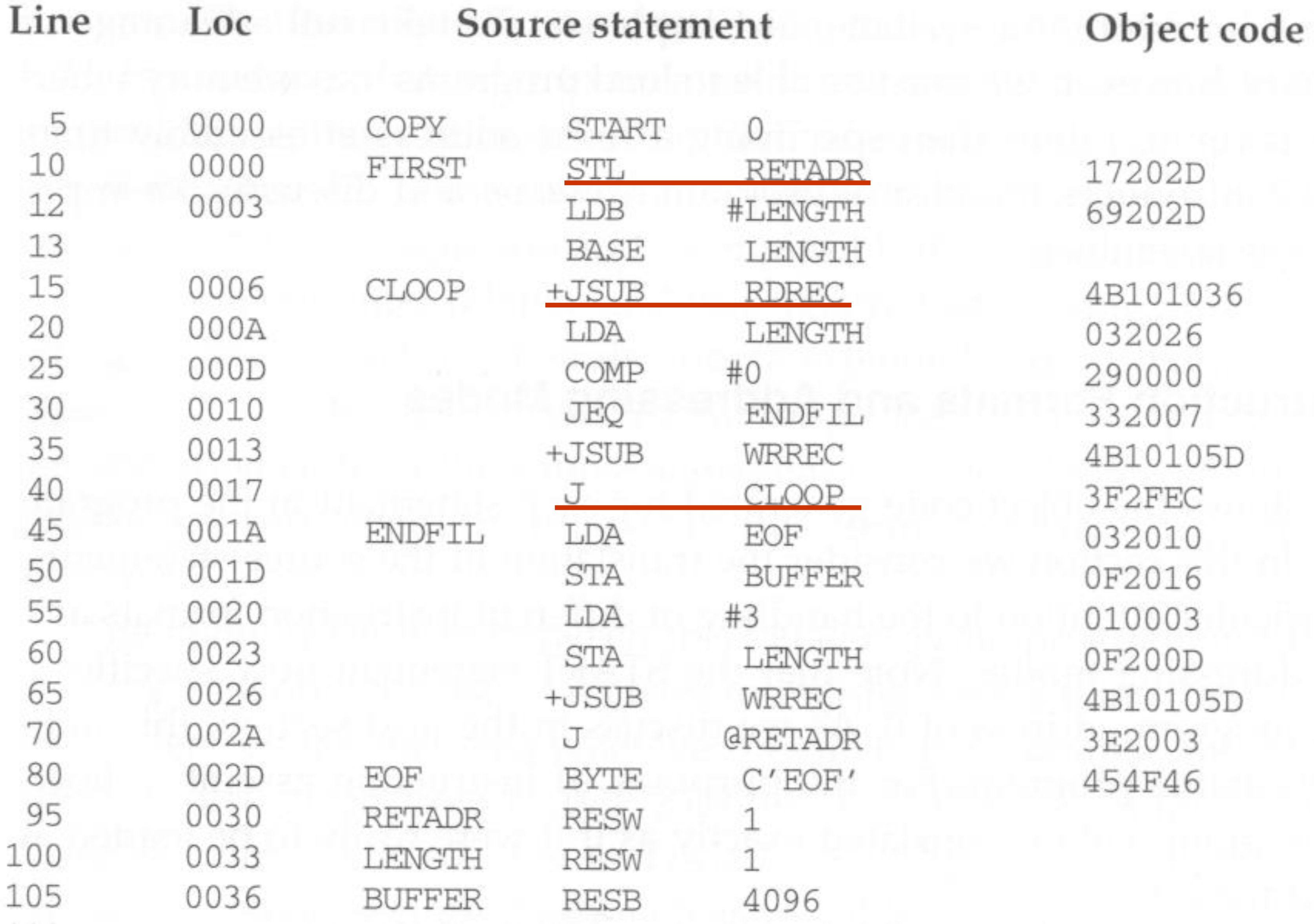
Considering Addressing Modes

- PC or base relative addressing
  - Calculate displacement
  - Displacement must be small enough to fit in the 12-bit field (-2048..2047 for PC relative mode, 0..4095 for base relative mode)
- Extended instruction format (4-byte)
  - 20-bit field for direct addressing

How Assembler Recognizes the Addressing Mode

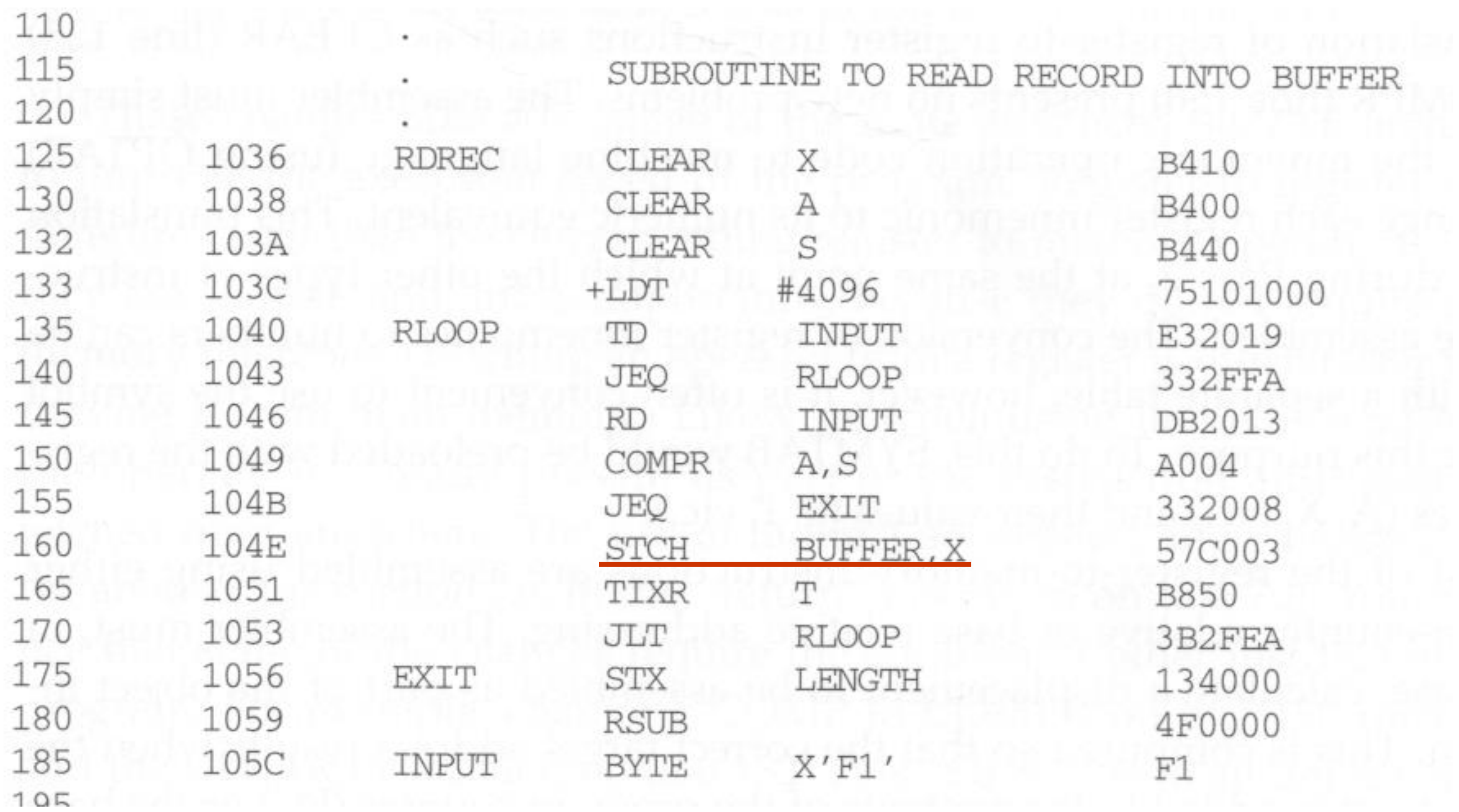
- Extended format: +op m
- Indirect addressing: op @m
- Immediate addressing: op #c
- Index addressing: op m,X
- Relative addressing: op m
  - 1st choice: PC relative (arbitrarily chosen)
  - 2nd choice: base relative (if displacement is invalid in PC relative mode)
  - 3rd choice: error message (if displacement is invalid in both relative modes)

SIC/XE Assembly with Object Code(fig.2.6)



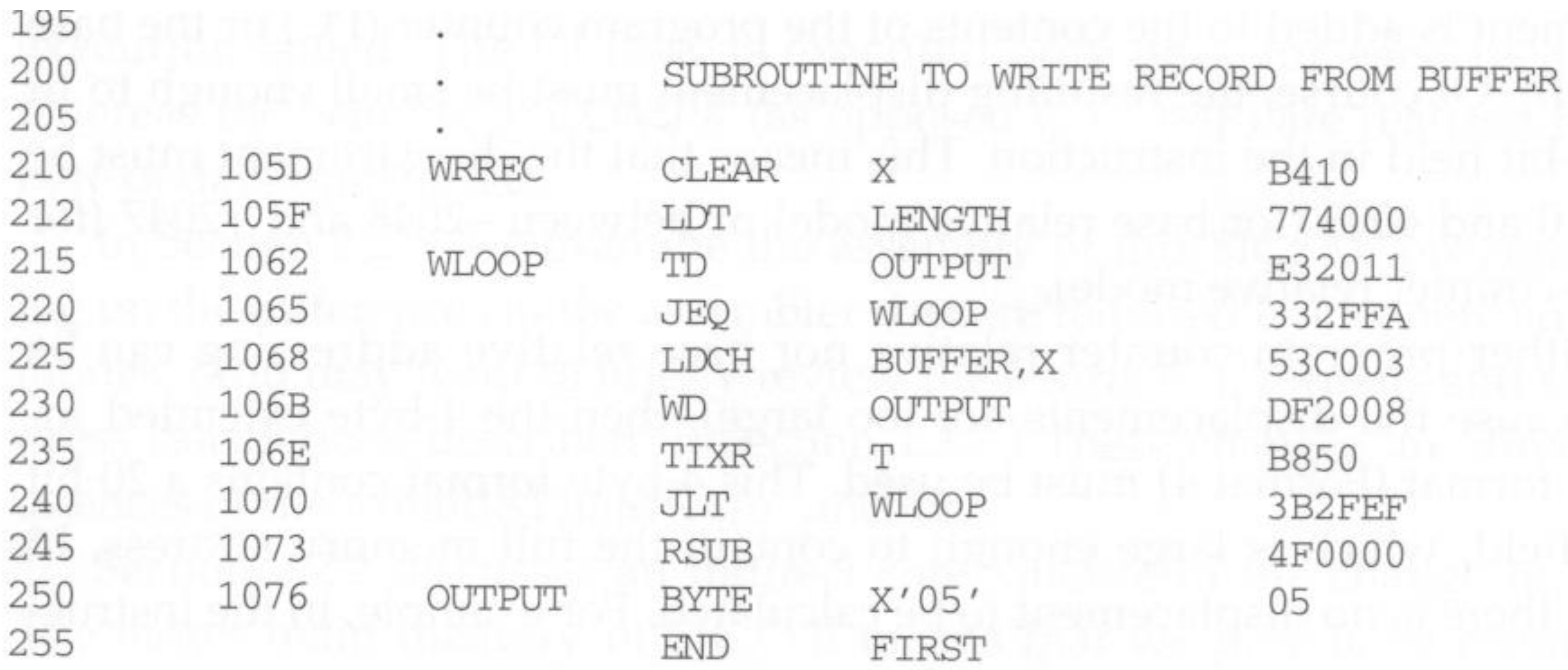
Line	Loc	Source statement			Object code
5	0000	COPY	START	0	
10	0000	FIRST	STL	RETADR	17202D
12	0003		LDB	#LENGTH	69202D
13			BASE	LENGTH	
15	0006	CLOOP	+JSUB	RDREC	4B101036
20	000A		LDA	LENGTH	032026
25	000D		COMP	#0	290000
30	0010		JEQ	ENDFIL	332007
35	0013		+JSUB	WRREC	4B10105D
40	0017		J	CLOOP	3F2FEC
45	001A	ENDFIL	LDA	EOF	032010
50	001D		STA	BUFFER	0F2016
55	0020		LDA	#3	010003
60	0023		STA	LENGTH	0F200D
65	0026		+JSUB	WRREC	4B10105D
70	002A		J	@RETADR	3E2003
80	002D	EOF	BYTE	C'EOF'	454F46
95	0030	RETADR	RESW	1	
100	0033	LENGTH	RESW	1	
105	0036	BUFFER	RESB	4096	

SIC/XE Assembly with Object Code



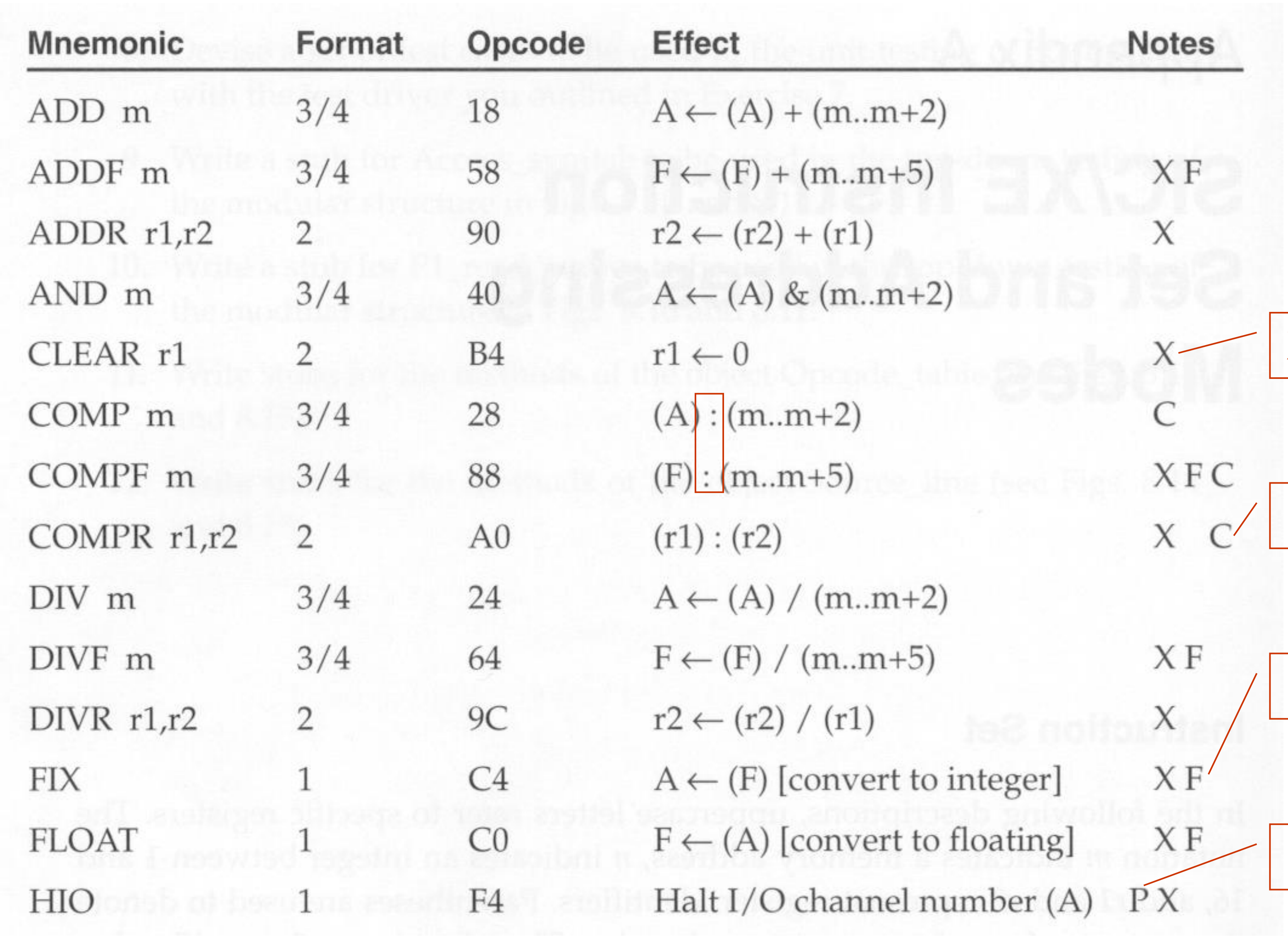
```
110              .
115              .          SUBROUTINE TO READ RECORD INTO BUFFER
120              .
125    1036      RDREC      CLEAR     X                  B410
130    1038                 CLEAR     A                  B400
132    103A                 CLEAR     S                  B440
133    103C                +LDT       #4096              75101000
135    1040      RLOOP      TD        INPUT              E32019
140    1043                 JEQ       RLOOP              332FFA
145    1046                 RD        INPUT              DB2013
150    1049                 COMPR     A,S                A004
155    104B                 JEQ       EXIT               332008
160    104E                 STCH      BUFFER,X           57C003
165    1051                 TIXR      T                  B850
170    1053                 JLT       RLOOP              3B2FEA
175    1056      EXIT       STX       LENGTH             134000
180    1059                 RSUB                         4F0000
185    105C      INPUT      BYTE      X'F1'              F1
```

SIC/XE Assembly with Object Code



```
195                .
200                .        SUBROUTINE TO WRITE RECORD FROM BUFFER
205                .
210     105D    WRREC       CLEAR    X                  B410
212     105F                LDT      LENGTH             774000
215     1062    WLOOP       TD       OUTPUT             E32011
220     1065                JEQ      WLOOP              332FFA
225     1068                LDCH     BUFFER,X           53C003
230     106B                WD       OUTPUT             DF2008
235     106E                TIXR     T                  B850
240     1070                JLT      WLOOP              3B2FEF
245     1073                RSUB                        4F0000
250     1076    OUTPUT      BYTE     X'05'              05
255                         END      FIRST
```

SIC/XE Instruction Set



Mnemonic	Format	Opcode	Effect	Notes
ADD m	3/4	18	A ← (A) + (m..m+2)	
ADDF m	3/4	58	F ← (F) + (m..m+5)	X F
ADDR r1,r2	2	90	r2 ← (r2) + (r1)	X
AND m	3/4	40	A ← (A) & (m..m+2)	
CLEAR r1	2	B4	r1 ← 0	X
COMP m	3/4	28	(A) : (m..m+2)	C
COMPF m	3/4	88	(F) : (m..m+5)	X F C
COMPR r1,r2	2	A0	(r1) : (r2)	X C
DIV m	3/4	24	A ← (A) / (m..m+2)	
DIVF m	3/4	64	F ← (F) / (m..m+5)	X F
DIVR r1,r2	2	9C	r2 ← (r2) / (r1)	X
FIX	1	C4	A ← (F) [convert to integer]	X F
FLOAT	1	C0	F ← (A) [convert to floating]	X F
HIO	1	F4	Halt I/O channel number (A)	P X

X: only for XE

C: set CC

F: floating-point

P: privileged

J m	3/4	3C	PC ← m	
JEQ m	3/4	30	PC ← m if CC set to =	
JGT m	3/4	34	PC ← m if CC set to >	
JLT m	3/4	38	PC ← m if CC set to <	
JSUB m	3/4	48	L ← (PC); PC ← m	
LDA m	3/4	00	A ← (m..m+2)	
LDB m	3/4	68	B ← (m..m+2)	X
LDCH m	3/4	50	A [rightmost byte] ← (m)	
LDF m	3/4	70	F ← (m..m+5)	X F
LDL m	3/4	08	L ← (m..m+2)	
LDS m	3/4	6C	S ← (m..m+2)	X
LDT m	3/4	74	T ← (m..m+2)	X
LDX m	3/4	04	X ← (m..m+2)	
LPS m	3/4	D0	Load processor status from information beginning at address m (see Section 6.2.1)	P X
MUL m	3/4	20	A ← (A) * (m..m+2)	

for interrupt

Mnemonic	Format	Opcode	Effect	Notes
MULF m	3/4	60	F ← (F) * (m..m+5)	X F
MULR r1, r2	2	98	r2 ← (r2) * (r1)	X
NORM	1	C8	F ← (F) [normalized]	X F
OR m	3/4	44	A ← (A) | (m..m+2)	
RD m	3/4	D8	A [rightmost byte] ← data from device specified by (m)	P
RMO r1,r2	2	AC	r2 ← (r1)	X
RSUB	3/4	4C	PC ← (L)	
SHIFTL r1,n	2	A4	r1 ← (r1); left circular shift n bits. {In assembled instruction, r2 = n–1}	X
SHIFTR r1,n	2	A8	r1 ← (r1); right shift n bits, with vacated bit positions set equal to leftmost bit of (r1). {In assembled instruction, r2 = n-1}	X
SIO	1	F0	Start I/O channel number (A); address of channel program is given by (S)	P X

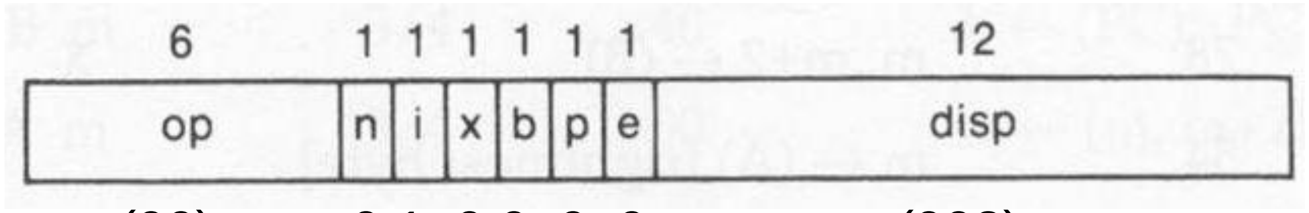
Set Storage Key for memory protection

SSK m	3/4	EC	Protection key for address m ← (A) (see Section 6.2.4)	P X
STA m	3/4	0C	m..m+2 ← (A)	
STB m	3/4	78	m..m+2 ← (B)	X
STCH m	3/4	54	m ← (A) [rightmost byte]	
STF m	3/4	80	m..m+5 ← (F)	X F
STI m	3/4	D4	Interval timer value ← (m..m+2) (see Section 6.2.1)	P X
STL m	3/4	14	m..m+2 ← (L)	
STS m	3/4	7C	m..m+2 ← (S)	X
STSW m	3/4	E8	m..m+2 ← (SW)	P
STT m	3/4	84	m..m+2 ← (T)	X
STX m	3/4	10	m..m+2 ← (X)	
SUB m	3/4	1C	A ← (A) − (m..m+2)	
SUBF m	3/4	5C	F ← (F) − (m..m+5)	X F

Mnemonic	Format	Opcode	Effect	Notes
SUBR r1,r2	2	94	r2 ← (r2) – (r1)	X
SVC n	2	B0	Generate SVC interrupt. {In assembled instruction, r1 = n}	X
TD m	3/4	E0	Test device specified by (m)	P C
TIO	1	F8	Test I/O channel number (A)	P X C
TIX m	3/4	2C	X ← (X) + 1; (X): (m..m+2)	C
TIXR r1	2	B8	X ← (X) + 1; (X): (r1)	X C
WD m	3/4	DC	Device specified by (m) ← (A) [rightmost byte]	P

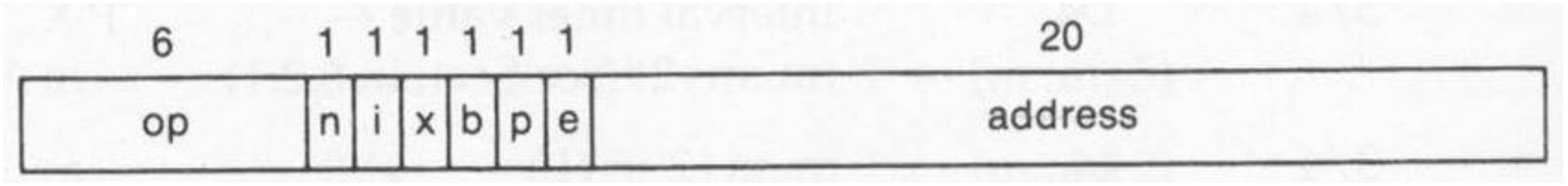
Immediate Addressing Mode

Instruction: 55 0020 LDA #3 010003



$$\underbrace{(00)_{16}\quad 0\ 1}_{(01)_{16}}\quad \underbrace{0\ 0\ \ 0\ \ 0}_{(0)_{16}}\qquad\qquad \underset{(003)_{16}}{(003)_{16}}$$

Instruction: 133 103C +LDT #4096 75101000



$$\underbrace{(74)_{16}\quad 0\ 1}_{(75)_{16}}\quad \underbrace{0\ 0\ \ 0\ \ 1}_{(1)_{16}}\qquad\qquad \underset{(01000)_{16}}{(01000)_{16}}$$

Immediate Addressing Mode

- LDA   #3

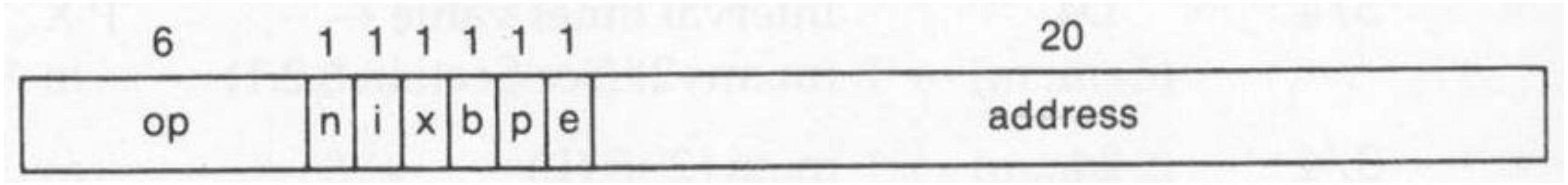
  ```
  ---- --ni  xbpe ---- ---- ----
  0000 0001  0000 0000 0000 0011 ➔ 010003
  ```

- +LDT   #4096

  ```
  ---- --ni  xbpe ----  ----  ----  ---- ----
  0111 0101  0001 0000  0001 0000  0000 0000
     ➔ 75101000
  ```

Extended Format

Instruction: 15 0006 CLOOP +JSUB RDREC 4B101036



$$\underbrace{(48)_{16}\quad 1\ 1}_{(4B)_{16}}\ \underbrace{0\ 0\ 0\ 1}_{(1)_{16}}\qquad \underset{(01036)_{16}}{(01036)_{16}}$$

Extended Format

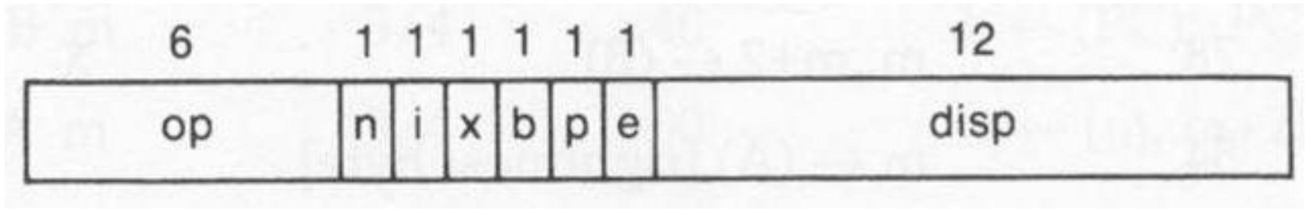
- +JSUB RDREC

```
 ----  --ni  xbpe ----  ----  ----  ---- ----
0100 1011  0001 0000  0001 0000  0011 0110
   ➔  4B101036
```

PC Relative Addressing Mode

Instruction:

```
10   0000   FIRST    STL    RETADR      17202D
12   0003            LDB    #LENGTH     69202D
 :    :
95   0030   RETADR   RESW   1
```



$$\underbrace{(14)_{16}\qquad 1\ 1}_{(17)_{16}}\quad \underbrace{0\ 0\ \ 1\ \ 0}_{(2)_{16}}\qquad \underset{(02D)_{16}}{(02D)_{16}}$$

PC is advanced after each instruction is fetched and before it is executed. That is, PC contains the address of the next instruction.

$$\text{disp} = (0030)_{16}-(0003)_{16} = (002D)_{16}$$

PC Relative Addressing Mode

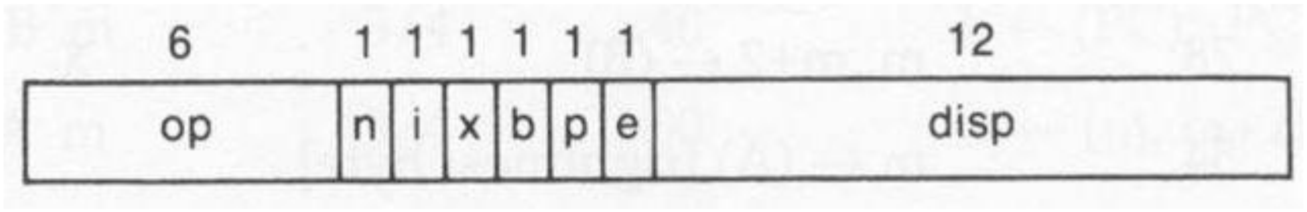
- STL RETADR

```
 - - - -  - - ni  xbpe ----  ---- ----
0001 0111  0010 ----  ---- ----

disp=RETADR-PC=0030-0003=002D
0001 0111  0010 0000  0010 1101 ➔ 17202D
```

PC Relative Addressing Mode

Instruction:	15	0006	CLOOP	+JSUB	RDREC	4B101036
	:	:				
	40	0017		J	CLOOP	3F2FEC
	45	001A	ENDFIL	LDA	EOF	032010



$$\underbrace{(3C)_{16}\quad 1\ 1}_{(3F)_{16}}\ \underbrace{0\ 0\ 1\ 0}_{(2)_{16}}\qquad \underbrace{(FEC)_{16}}_{(FEC)_{16}}$$

$$disp = (0006)_{16} - (001A)_{16} = (-14)_{16} = (FEC)_{16}$$

PC Relative Addressing Mode

- J CLOOP

```
- - - -  - - ni   xbpe ----  ---- ----
0011 1111  0010 ----  ---- ----
```

disp=CLOOP-PC=006-01A=FEC=$(-14)_{16}$

0011 1111 0010 1111 1110 1100 ➔ 3F2FEC

PC Relative Addressing Mode

- J CLOOP

```
---- --ni xbpe ---- ---- ----
0011 1111 0bp0 ---- ---- ----
```

CLOOP-PC=0006-001A=FEC

$$
\begin{array}{rl}
 & 0000\ 0001\ 0100=(14)_{16}=(20)_{10} \\
+ & 1111\ 1110\ 1100=(-14)_{16}=(-20)_{10} \\
\hline
 & 0000\ 0000\ 0000=(0)_{16}=(0)_{10}
\end{array}
$$

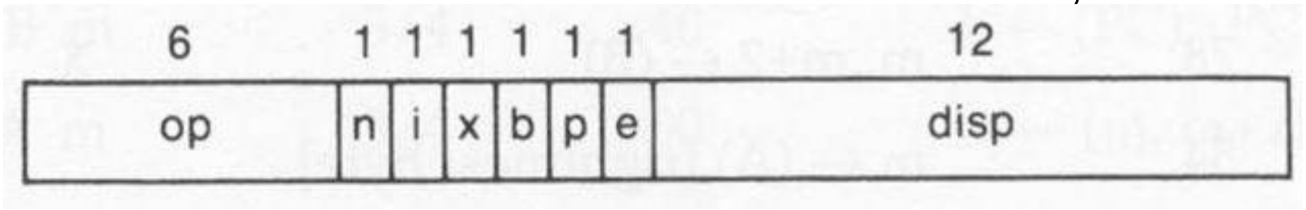
Decimal	Signed-magnitude	Signed-1's complement	Signed-2's complement
+7	0111	0111	0111
+6	0110	0110	0110
+5	0101	0101	0101
+4	0100	0100	0100
+3	0011	0011	0011
+2	0010	0010	0010
+1	0001	0001	0001
+0	0000	0000	0000
-0	1000	1111	X
-1	1001	1110	1111
-2	1010	1101	1110
-3	1011	1100	1101
-4	1100	1011	1100
-5	1101	1010	1011
-6	1110	1001	1010
-7	1111	1000	1001
-8	X	X	1000

0011 1111 0010 1111 1110 1100 $\rightarrow$ 3F2FEC

Base Relative Addressing Mode

Instruction:

```
12    0003                  LDB    #LENGTH      69202D
13                          BASE   LENGTH
 :      :
100   0033   LENGTH   RESW   1
105   0036   BUFFER   RESB   4096
 :      :
160   104E            STCH   BUFFER,X     57C003
```



$$\underbrace{(54)_{16}\quad 1\ 1}_{(57)_{16}}\quad \underbrace{1\ 1\ 0\ \ 0}_{(C)_{16}}\qquad \underbrace{(003)_{16}}_{(003)_{16}}$$

- PC relative is no longer applicable
- BASE directive explicitly informs the assembler that the base register will contain the address of LENGTH (use NOBASE to invalidate)
- LDB loads the address of LENGTH into base register during execution

$$\text{disp} = (0036)_{16} - (0033)_{16} = (0003)_{16}$$

Base Relative Addressing Mode

- STCH   BUFFER,X

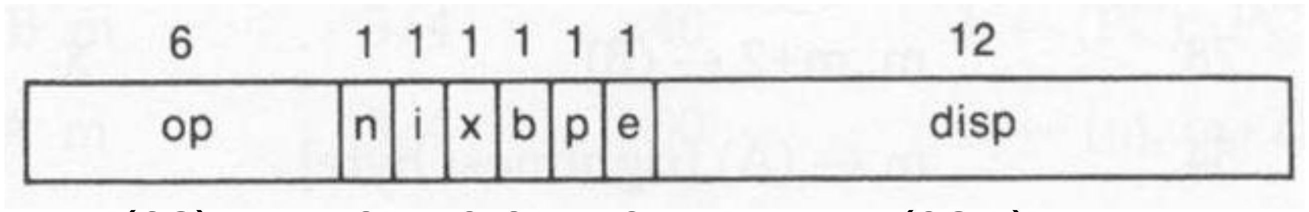
```
- - - -  - - ni   xbpe ----  ---- ----
0101 0111  1bp0 ----  ---- ----


BUFFER-PC=0036-1051
   → PC-relative fails
BUFFER-BASE=0036-0033=003


0101 0111  1100 0000  0000 0011 → 57C003
```

Immediate + PC Relative Addressing Mode

```
Instruction: 12     0003                   LDB     #LENGTH          69202D
             13                            BASE     LENGTH
             15     0006    CLOOP  +JSUB     RDREC            4B101036
              :        :
             100    0033   LENGTH   RESW   1
```



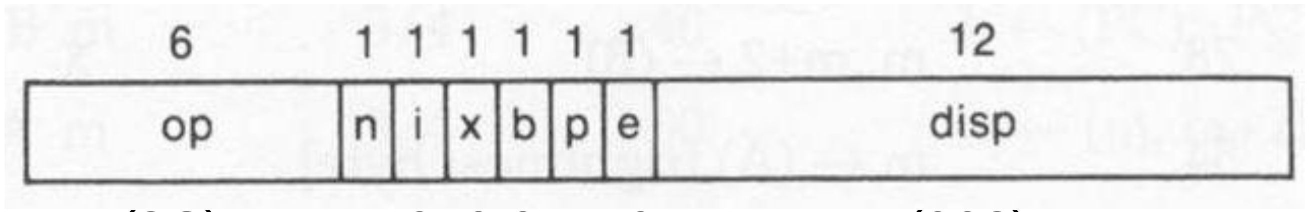
$$\underbrace{(68)_{16}\quad 0\ 1}_{(69)_{16}}\quad \underbrace{0\ 0\ \ 1\ \ 0}_{(2)_{16}}\qquad\qquad (02D)_{16}$$

$$\qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad (02D)_{16}$$

disp = $(0033)_{16}$-$(0006)_{16}$ = $(002D)_{16}$

Indirect + PC Relative Addressing Mode

```
Instruction: 70    002A              J         @RETADR          3E2003
             80    002D   EOF        BYTE        C'EOF'         454F46
             95    0030   RETADR    RESW   1
```



$$\underbrace{(3C)_{16}\qquad 1\ 0}_{(3E)_{16}}\quad \underbrace{0\ 0\ \ 1\ \ 0}_{(2)_{16}}\qquad\qquad \underset{(003)_{16}}{(003)_{16}}$$

$$\text{disp} = (0030)_{16}-(002D)_{16} = (0003)_{16}$$

Why Program Relocation

- To increase the productivity of the machine

↓

- Want to load and run several programs at the same time (multiprogramming)

↓

- Must be able to load programs into memory wherever there is room

↓

- Actual starting address of the program is not known until load time

Absolute Program

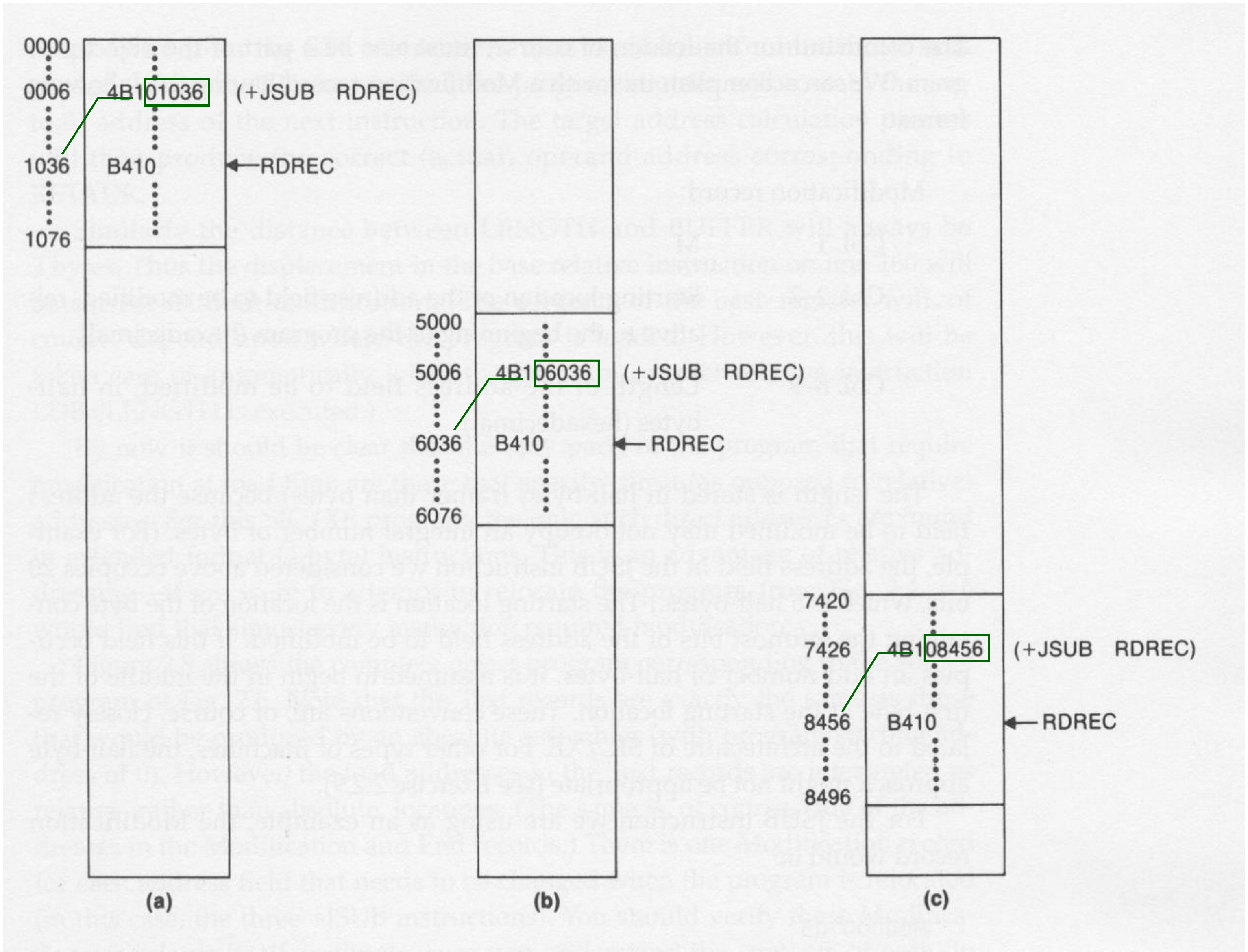
- Program with starting address specified at assembly time
- In the example of SIC assembly program

```
Instruction:   55     101B              LDA    THREE              00102D
```

Calculated from the starting address 1000

- The address may be invalid if the program is loaded into some where else.

Relocatable Program



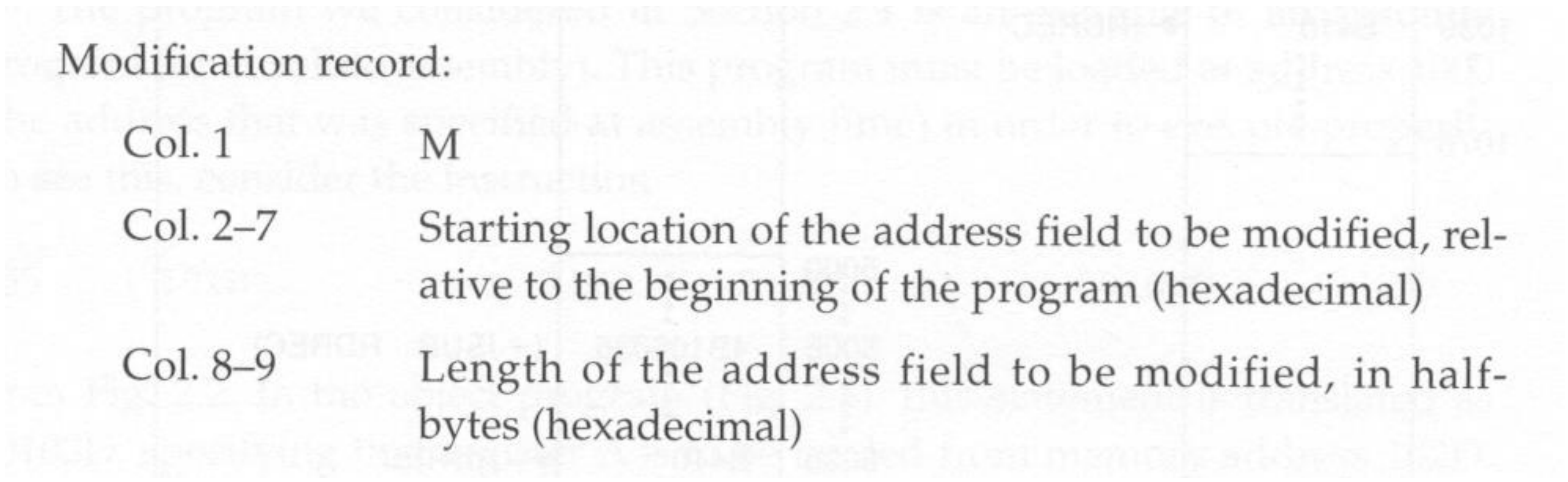
What Needs to be Relocated

- Need to be modified:
  - The address portion of those instructions that use absolute (direct) addresses.
- Need not be modified:
  - Register-to-register instructions (no memory references)
  - PC or base-relative addressing (relative displacement remains the same regardless of different starting addresses)

How to Relocate Addresses

- For Assembler
  - For an address label, its address is assigned relative to the start of the program (that's why START 0)
  - Produce a modification record to store the starting location and the length of the address field to be modified.
- For loader
  - For each modification record, add the actual beginning address of the program to the address field at load time.

Format of Modification Record

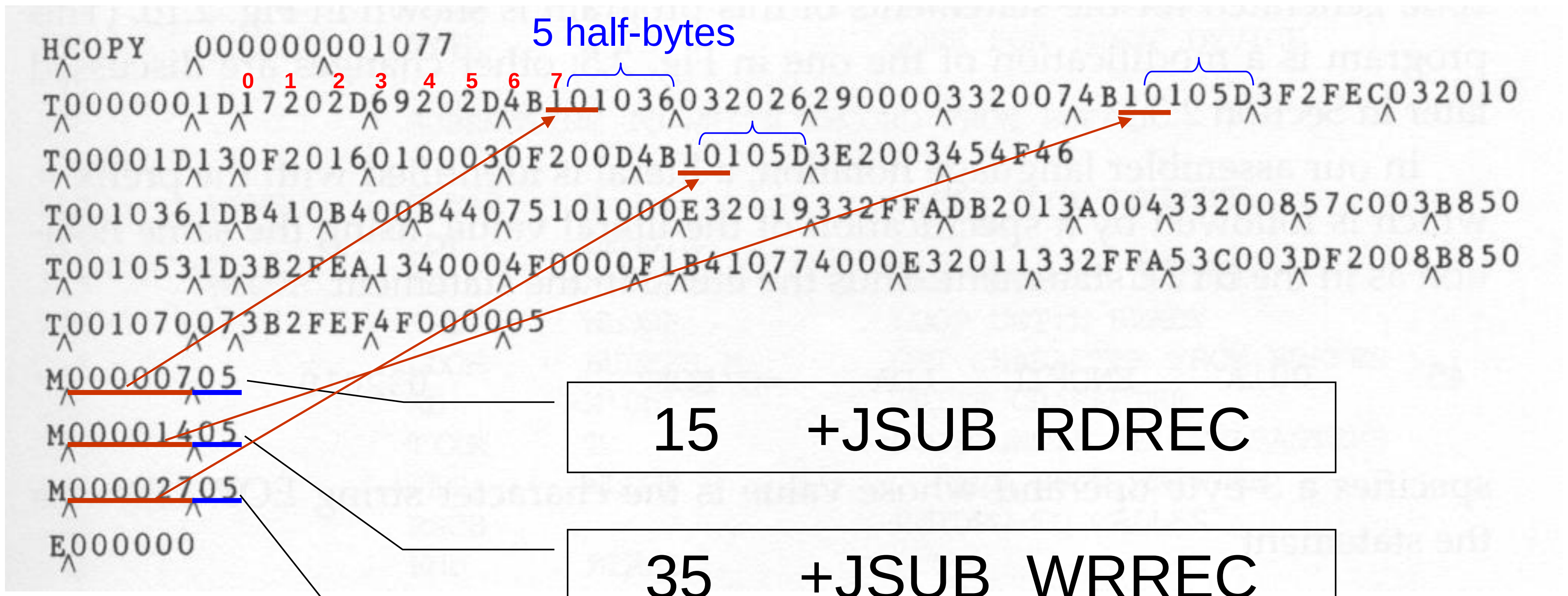


Modification record:

Col. 1	M
Col. 2–7	Starting location of the address field to be modified, relative to the beginning of the program (hexadecimal)
Col. 8–9	Length of the address field to be modified, in half-bytes (hexadecimal)

- One modification record for each address to be modified
- The length is stored in half-bytes (20 bits = 5 half-bytes)
- The starting location is the location of the byte containing the leftmost bits of the address field to be modified.
- If the field contains an odd number of half-bytes, the starting location begins in the middle of the first byte.

Relocatable Object Program



```
H^COPY  ^000000^001077
T^000000^1D^17202D^69202D^4B101036^032026^290000^332007^4B10105D^3F2FEC^032010
T^00001D^13^0F2016^010003^0F200D^4B10105D^3E2003^454F46
T^001036^1D^B410^B400^B440^75101000^E32019^332FFA^DB2013^A004^332008^57C003^B850
T^001053^1D^3B2FEA^134000^4F0000^F1^B410^774000^E32011^332FFA^53C003^DF2008^B850
T^001070^07^3B2FEF^4F0000^05
M^000007^05
M^000014^05
M^000027^05
E^000000
```

15 +JSUB RDREC

35 +JSUB WRREC

65 +JSUB WRREC